PATENT ABSTRACTS OF JAPAN

(11)Publication number : **09-036747**
(43)Date of publication of application : **07.02.1997**

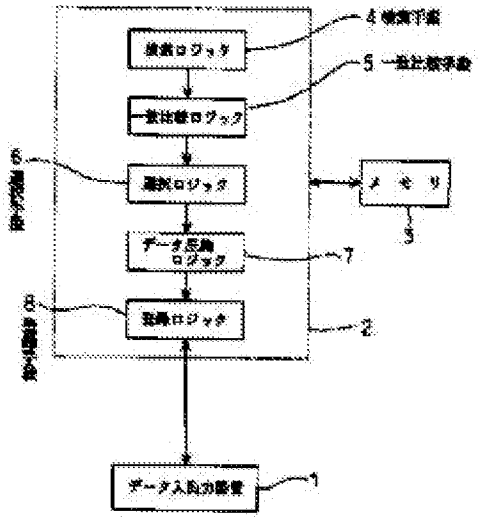
(51)Int.Cl. H03M 7/40
G06F 5/00
G06F 17/30

(21)Application number : **07-181485**
(22)Date of filing : **18.07.1995**

(71)Applicant : **TOSHIBA CORP**
TOSHIBA COMPUT ENG CORP
(72)Inventor : **SAKAKIBARA HIDETOSHI**
TODA ATSUKO

(54) DATA COMPRESSION METHOD AND DATA COMPRESSOR

(57)Abstract:
PROBLEM TO BE SOLVED: To provide the data compression method and the data compressor by which data can be compressed at a high speed without taking mush time for data retrieval by adopting the hash method for data compression by the LZ slide dictionary method so as to retrieve data.
SOLUTION: When a data input output device 1 reads compression processing object data to a memory 3, a control circuit 2 retrieves the data string the same as compression processing object data string from a hash table by the hash method and compares the data of the retrieved data string with that of the compression processing object data string one by one and provides an output of equal data length of both the data strings, repeats the retrieval and comparison till the retrieval object is not in existence and selects a data string whose equal data length is longest among data strings at the retrieved hand position and succeeding positions and provides an output of the head position of the selected data string and the equal data length as equality information and registers the head position of the compression processing object data string to a hash table as a retrieval object.



(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

特開平9－36747

(43)公開日　平成９年(1997)２月７日

(51)Int.Cl.[6]	識別記号	庁内整理番号	ＦＩ		技術表示箇所
Ｈ０３Ｍ　7/40		9382－5K	Ｈ０３Ｍ	7/40	
Ｇ０６Ｆ　5/00			Ｇ０６Ｆ	5/00	Ｈ
17/30		9289－5L		15/411	３１０

審査請求　未請求　請求項の数10　ＯＬ　（全 13 頁）

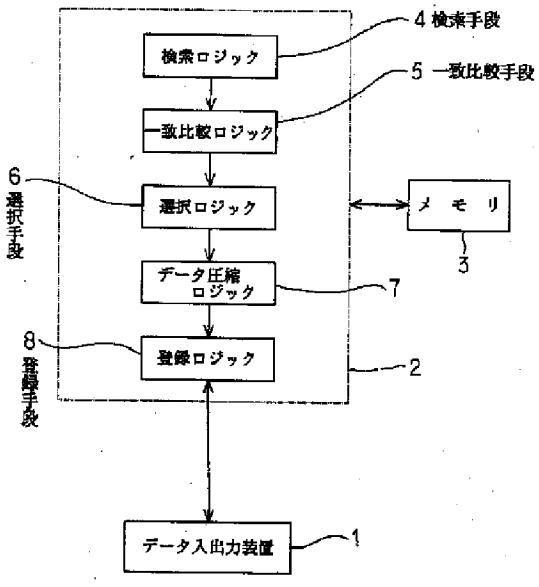
(21)出願番号　特願平7－181485

(22)出願日　平成７年(1995)７月18日

(71)出願人　000003078
株式会社東芝
神奈川県川崎市幸区堀川町72番地
(71)出願人　000221052
東芝コンピュータエンジニアリング株式会社
東京都青梅市新町1381番地１
(72)発明者　榊原　秀敏
東京都青梅市新町1381番地１　東芝コンピュータエンジニアリング株式会社内
(72)発明者　戸田　亜津子
名古屋市西区名西２丁目33番10号　株式会社東芝中部システムセンター内
(74)代理人　弁理士　佐藤　強

(54)【発明の名称】　データ圧縮方法及びデータ圧縮装置

(57)【要約】
【課題】　ＬＺスライド辞書法によるデータ圧縮においてハッシュ法を用いてデータ検索を行うことにより、データ検索に時間を要すること無く、高速でデータの圧縮を行うことができるデータ圧縮方法及びデータ圧縮装置を提供する。
【解決手段】　データ入出力装置１によって圧縮処理対象データをメモリ３上に読込むと、制御回路２は、圧縮処理対象データ列と同じデータ列をハッシュ法によってハッシュ表より検索し、その検索されたデータ列と圧縮処理対象データ列との各データを１データずつ比較して両データ列の一致データ長を出力する。検索候補がなくなるまで検索及び比較を繰返して、出力された一致データ長が最長であるデータ列を検索された先頭位置から始まるデータ列の中から選択する。そして、選択されたデータ列の先頭位置と一致データ長とを一致情報として出力し、圧縮処理対象データ列の先頭位置を検索候補としてハッシュ表に登録するよう構成した。



1

【特許請求の範囲】

【請求項１】　ＬＺスライド辞書法を用いて文書データ等のデータを圧縮するデータ圧縮方法において、

圧縮処理対象のデータ中における圧縮処理済みデータの次の位置から始まる圧縮対象データ列と同じデータ列の候補を前記圧縮処理済みデータの中からハッシュ法によって検索する検索ステップと、

この検索ステップにおいて検索されたデータ列と、前記圧縮対象データ列とを比較して一致しているデータ長を求める比較ステップと、

前記圧縮対象データ列と同じデータ列の候補を前記圧縮処理済みデータの中から検索候補がなくなるまで前記検索ステップと前記比較ステップを繰返し実行する繰返しステップと、

前記比較ステップにおいて求められた一致データ長の中から最長のデータ長のデータ列を選択して出力する選択ステップと、

前記圧縮対象データ列をハッシュ法による検索候補としてハッシュ表に登録する登録ステップとを備えたことを特徴とするデータ圧縮方法。

【請求項２】　前記繰返しステップにおいては、検索ステップ及び比較ステップを所定回数繰返したら、その後の検索ステップ及び比較ステップを打切るように構成されていることを特徴とする請求項１記載のデータ圧縮方法。

【請求項３】　前記繰返しステップにおける打切り回数は、３乃至２０回に設定されていることを特徴とする請求項２記載のデータ圧縮方法。

【請求項４】　前記登録ステップにおいては、前記圧縮対象データ列をハッシュ法による検索候補のうちの最初に検索される位置に登録することを特徴とする請求項１乃至３の何れかに記載のデータ圧縮方法。

【請求項５】　前記比較ステップにおいて最長の一致データ長を検出したときの検索回数を統計的に記憶することにより、前記繰返しステップにおける打切り回数を自動的に最適化して決定することを特徴とする請求項２または４記載のデータ圧縮方法。

【請求項６】　ＬＺスライド辞書法を用いて文書データ等のデータを圧縮するデータ圧縮装置において、

圧縮処理対象のデータ中における圧縮処理済みデータの次の位置から始まる圧縮対象データ列と同じデータ列の候補を前記圧縮処理済みデータの中からハッシュ法によって検索する検索手段と、

この検索手段により検索されたデータ列と、前記圧縮対象データ列とを比較して一致しているデータ長を求める比較手段と、

前記圧縮対象データ列と同じデータ列の候補を前記圧縮処理済みデータの中から検索候補がなくなるまで検索及び比較を繰返し実行する繰返し手段と、

前記比較手段により求められた一致データ長の中から最長のデータ長のデータ列を選択して出力する選択手段と、

前記圧縮対象データ列をハッシュ法による検索候補としてハッシュ表に登録する登録手段とを備えたことを特徴とするデータ圧縮装置。

【請求項７】　前記繰返し手段は、検索及び比較を所定回数繰返したら、その後の検索及び比較を打切るように構成されていることを特徴とする請求項６記載のデータ圧縮装置。

【請求項８】　前記繰返し手段における打切り回数は、３乃至２０回に設定されていることを特徴とする請求項７記載のデータ圧縮装置。

【請求項９】　前記登録手段は、前記圧縮対象データ列をハッシュ法による検索候補のうちの最初に検索される位置に登録することを特徴とする請求項６乃至８の何れかに記載のデータ圧縮装置。

【請求項１０】　前記繰返し手段は、前記比較手段により最長の一致データ長を検出したときの検索回数を統計的に記憶することにより、検索及び比較の打切り回数を自動的に最適化して決定することを特徴とする請求項６または９記載のデータ圧縮装置。

【発明の詳細な説明】

【０００１】

【発明が属する技術分野】本発明は、文章データ等のデータを圧縮するデータ圧縮方法及びデータ圧縮装置に関する。

【０００２】

【従来の技術】文章データ等のデータを圧縮するデータ圧縮方法の一例として、ＬＺ（Lempel-Ziv）スライド辞書法がある。このＬＺスライド辞書法は、図１２に示すように、圧縮処理対象であるデータをその先頭から検索していき、１度出現したデータ列が再度出現した場合は、以前に出現したデータ列の先頭アドレスと両データ列で一致するデータの長さとを一致情報として、再度出現したデータ列に置換えて出力することによりデータ圧縮を行う方法である。

【０００３】このＬＺスライド辞書法においては、圧縮対象データ列が以前に出現したデータであるか否かを検索するために、様々なデータの検索法が用いられている。例えば、２分木による検索法においては、入力データ中に現れる各文字コードを先頭（根節点）として、順次２つの節点に分岐して行く木構造をなす検索パスを構成する。そして、ある文字コードとその後に続く文字コードとについて、これらの文字コードを例えば１６進数コードで表現した場合の大小の比較結果によって、後の文字コードをどちらの節点に分岐させて配置するかを決定して行く。

【０００４】

【発明が解決しようとする課題】しかしながら、上記２分木による検索では、一つのデータ列を検索するのに節

3

点に配置された一文字毎に比較していくため、木の成長状態がアンバランスになっている（検索パスの一部だけが長くなっている）場合には、その検索パスの最終の節点（葉節点）に配置されたデータを検索するには、非常に長い時間を要するという欠点がある。

【０００５】また、トライ木による検索法が採用されることもあり、このトライ木の場合には、２分木に比べてより高速に検索を行うことができる。しかし、上記トライ木の場合、膨大なメモリ空間を必要とするという不具合があった。

【０００６】本発明は上記課題を解決するもので、その目的は、ＬＺスライド辞書法によるデータを圧縮する場合に、データの検索を高速で実行できると共に、検索時に必要なメモリ空間を極力少なくすることができるデータ圧縮方法及びデータ圧縮装置を提供するにある。

【０００７】

【課題を解決するための手段】本発明のデータ圧縮方法は、ＬＺスライド辞書法を用いて文書データ等のデータを圧縮するデータ圧縮方法において、圧縮処理対象のデータ中における圧縮処理済みデータの次の位置から始まる圧縮対象データ列と同じデータ列の候補を前記圧縮処理済みデータの中からハッシュ法によって検索する検索ステップと、この検索ステップにおいて検索されたデータ列と前記圧縮対象データ列とを比較して一致しているデータ長を求める比較ステップと、前記圧縮対象データ列と同じデータ列の候補を前記圧縮処理済みデータの中から検索候補がなくなるまで前記検索ステップと前記比較ステップを繰返し実行する繰返しステップと、前記比較ステップにおいて求められた一致データ長の中から最長のデータ長のデータ列を選択して出力する選択ステップと、前記圧縮対象データ列をハッシュ法による検索候補としてハッシュ表に登録する登録ステップとを備えたところに特徴を有する。

【０００８】この場合、繰返しステップにおいては、検索及び比較を所定回数繰返した後打切るように構成することが好ましい。また、検索及び比較の打切り回数は、３乃至２０回に設定することが好ましい。更に、登録ステップにおいては、圧縮対象データ列をハッシュ法による検索候補のうちの最初に検索される位置に登録するように構成することが良い。加えて、比較ステップにおいて最長の一致データ長を検出したときの検索回数を統計的に記憶することにより、繰返しステップにおける打切り回数を自動的に最適化して決定するように構成しても良い。

【０００９】本発明のデータ圧縮装置は、ＬＺスライド辞書法を用いて文書データ等のデータを圧縮するデータ圧縮装置において、圧縮処理対象のデータ中における圧縮処理済みデータの次の位置から始まる圧縮対象データ列と同じデータ列の候補を前記圧縮処理済みデータの中からハッシュ法によって検索する検索手段と、この検索

4

手段により検索されたデータ列と前記圧縮対象データ列とを比較して一致しているデータ長を求める比較手段と、前記圧縮対象データ列と同じデータ列の候補を前記圧縮処理済みデータの中から検索候補がなくなるまで検索及び比較を繰返し実行する繰返し手段と、前記比較手段により求められた一致データ長の中から最長のデータ長のデータ列を選択して出力する選択手段と、前記圧縮対象データ列をハッシュ法による検索候補としてハッシュ表に登録する登録手段とを備えたところに特徴を有する。

【００１０】この構成の場合、繰返し手段は、検索及び比較を所定回数繰返したら、その後の検索及び比較を打切るように構成することが良い。また、繰返し手段における打切り回数は、３乃至２０回の何れかに設定することが好ましい。更に、登録手段を、圧縮対象データ列をハッシュ法による検索候補のうちの最初に検索される位置に登録するように構成することも良い。そして、比較手段により最長の一致データ長を検出したときの検索回数を統計的に記憶することにより、繰返し手段における打切り回数を自動的に最適化して決定するように構成することも好ましい構成である。

【００１１】上記手段によれば、圧縮対象データ列と同じデータ列を圧縮処理済みデータの中からハッシュ法によって検索するから、データ列を高速で検索することができる。そして、このハッシュ法による検索時には、メモリ空間にハッシュ表を作成するが、このハッシュ表には、検索候補のデータ列の位置（圧縮処理済みデータ中における位置）を登録するだけであり、実際のデータを登録しないから、ハッシュ表はそれほど大きなメモリ空間を使用しない。また、ハッシュチェーンも、検索候補のデータ列の位置をチェーン状に登録するだけであり、実際のデータを登録しないから、それほど大きなメモリ空間を使用しない。

【００１２】また、検索及び比較を所定回数繰返した後、打切るように構成すると、データの検索をより一層高速に実行できる。更に、検索及び比較を打切る回数を、３乃至２０回に設定すると、データの圧縮効率を十分良好なものとしながら、圧縮処理に要する時間も十分短くすることができる。加えて、圧縮対象データ列をハッシュ法による検索候補のうちの最初に検索される位置に登録するように構成すると、最長の一致データ長を早く得る確率を高くすることができる。また、比較ステップにおいて、最長の一致データ長を検出したときの検索回数を統計的に記憶することにより、繰返しステップにおける打切り回数を自動的に最適化して決定するように構成すると、データ圧縮効率及び圧縮処理速度を最適化することができる。

【００１３】

【発明の実施の形態】以下本発明の第１実施例について、図１乃至図１０を参照して説明する。データ圧縮装

5

置の電気的構成を示す図１において、データの入出力手段であるデータ入出力装置１は例えばＳＣＳＩインターフェイスなどからなり、圧縮対象となるデータファイルが記憶されている例えばハードディスクやフロッピーディスクなどの図示しない外部記憶装置に対してデータの入出力を行う機能を有している。上記データ入出力装置１は、マイコンなどからなる制御回路２とアドレス及びデータバスライン並びに制御信号線を介して接続されている。上記制御回路２には、メモリ３がアドレス及びデータバスライン並びに制御信号線を介して接続されている。

【００１４】また、制御回路２には、データ圧縮処理を実行制御するための制御プログタムが記憶されている。そして、上記制御回路２は、検索手段である検索ロジック４、比較手段である一致比較ロジック５、選択手段である選択ロジック６、データ圧縮ロジック７、並びに、登録手段である登録ロジック８としての各機能を有するように構成されている。そして、制御回路２は、繰返し手段としての機能をも有している。以下、上記各ロジックの機能について説明する。まず、検索ロジック４は、メモリ３上に読込まれた圧縮対象データファイル中の圧縮対象データ列に対してハッシング（ハッシュ関数による演算）を行い、その圧縮対象データ列と同じデータ列を、後述する検索表に先頭位置を示す位置情報が書込まれて登録されているデータ列の中から検索する機能を有している。

【００１５】一致比較ロジック５は、圧縮対象データ列と、検索ロジック４によって検索された先頭位置から始まるデータ列とを１データ（例えば１文字）ずつ比較して、両者で一致しているデータ長を出力する機能を有している。選択ロジック６は、一致比較ロジック５によって出力される一致データ長が最長であるデータ列を、検索ロジック４により検索された先頭位置から始まるデータ列の中から選択する機能を有している。データ圧縮ロジック７は、選択ロジック６によって選択されたデータ列の先頭位置と、一致比較ロジック５がそのデータ列について出力した一致データ長とを一致情報として出力するものである。そして、登録ロジック８は、圧縮処理対象データ列の先頭位置を検索候補である位置情報としてハッシュ表に登録するものである。

【００１６】次に、上記実施例の作用について図２乃至図１０を参照して説明する。尚、図２及び図３に示すフローチャートは、制御回路２の制御プログラムの制御内容、即ち、データ圧縮処理の概略制御内容を示すものである。また、圧縮対象データ（ファイル）として、図４（ａ）に示すような文字を並べて成るデータを用いる。この文字列のデータをデータ圧縮装置によって圧縮する場合、操作者はデータ圧縮装置の図示しないキーボードなどの入力手段を操作することにより、圧縮対象データファイルのファイル名を指定すると共に、圧縮処理の実

6

行コマンドを入力する。すると、制御回路２は、上記指定されたデータファイルをデータ入出力装置１を介して外部記憶装置から読込んでメモリ３上に転送した後、データ圧縮処理を開始する。

【００１７】まず、図２のフローチャートにおいて、「初期処理」の処理ステップＲ１を実行する。ここでは、制御回路２は、メモリ３における圧縮処理に使用する領域のの初期化などを行う。続いて、「検索処理」の処理ステップＲ２に移行する。この「検索処理」は、図３のフローチャートによって示されるサブルーチンとして定義されており、以下この図３のフローチャートに従って説明する。

【００１８】図３に示すように、制御回路２は、メモリ３上に設けられた検索処理に関する領域の初期化を行い（ステップＳ１）、続いて、次の「データ検索」を実行する（ステップＳ２）。ここでは、検索ロジック４によって、ハッシングを行うためにメモリ３より図４（ａ）に示すデータ列の０番目（第０アドレス）の文字データ「Ａ」と、１番目（第１アドレス）の文字データ「Ｂ」の２文字を読出す。この場合、ハッシュ関数として排他的論理和を選び、第０アドレスの文字データ「Ａ」のＪＩＳコード**0x41**と、第１アドレスの文字データ「Ｂ」のＪＩＳコード**0x42**との排他的論理和を論理演算して、ハッシュ値**0x03**を得る。そして、このハッシュ値**0x03**によってハッシュ表の検索を行うように構成されている。

【００１９】上記ハッシュ表はハッシュ値をインデックスとして、そのハッシュ値を持つ文字列の先頭位置（アドレス）が位置情報として登録されている表（データテーブル）であり、メモリ３上に展開されている。尚、前記ステップＲ１の初期化において、ハッシュ表及び後述するハッシュチェーンには例えば全ビット「１」のデータ（**0xFFFF**…）が書込まれるように構成されており、これにより、データの登録が無い状態となっている。従って、この時点でハッシュ値**0x03**によってハッシュ表のデータ検索を行って登録されているデータを読出すと、”登録データなし”を検出する（図５参照）。尚、ハッシュ表とハッシュチェーンとを合わせたものを、検索表（検索対象）として定義する。

【００２０】続いて、「登録データなしか？」の判断を行う（ステップＳ３）。この時点では”登録データなし”を検出しているので、ステップＳ３にて「ＹＥＳ」へ進み、図２に示すステップＲ３へ移行し、「検索結果出力」の処理を実行する。ここでは、データ圧縮ロジック７によって、ステップＲ２で行われた検索処理の結果を、出力データ列として図４（ｂ）に示すような形式で出力するものである。尚、図４（ｂ）に示すように、圧縮処理された出力データ列の第０アドレスには管理部が配置されるが、この管理部の設定については後述する。

【００２１】また、上記ステップＲ３においては、ステップＲ２の検索処理の結果、最長一致データ長が所定値

7

（例えば「３」）以上か否かを判断してデータを出力する。具体的には、この時点では、ステップＳ３で登録データを検出できなかったので一致情報の出力は行われず、最長一致データ長は初期化されたまま「０」であり、所定値「３」未満であるから、第１アドレスに文字データ「Ａ」をそのまま出力する。そして、「ハッシュ表への登録」の処理を実行する（ステップＲ４）。ここでは、ハッシュ表への位置情報の登録を行う。この登録処理は、図６乃至１０に示す概念図のようにして実行される。

【００２２】具体的には、この時点での検索結果は、ハッシュ表のインデックス0x03に対して登録されているデータは一つも無かったので、ハッシュ表に先頭のデータ「Ａ」の位置（アドレス）を示す位置情報「０」を、”登録データなし”を示すコードに上書きして登録する（図６参照）。そして、圧縮対象データ列のポインタをインクリメントして、検索ロジック４が次に第１アドレスの文字データ「Ｂ」に注目するようにする。また、ループカウンタをインクリメントする。

【００２３】続いて、「処理したデータ数分ループしたか？」の判断を行う（ステップＲ５）。ここでは、処理データ数とループカウンタのカウント値とを比較することにより、ステップＲ４におけるハッシュ表への登録処理を処理データ数分ループして実行したか否かを判断する。ここで、処理データ数は、選択ロジック６によって得られる最長一致データ長が「３」以上の場合、後述するように圧縮処理が行われるため、その一致データ長に等しく設定される数値である。そして、最長一致データ長が「０」または「２」の場合は圧縮処理は行われず、１文字データがそのまま出力されるので、処理データ数は「１」と設定される数値である（最長一致データ長「１」は出力されない）。

【００２４】そして、この時点では、最長一致データ長は「０」であって処理データ数は「１」に設定されているから、ループカウンタのカウント値「１」に等しく、判断ステップＲ５において「ＹＥＳ」へ進み、「データ終わり？」の判断を行う（ステップＲ６）。ここでは、圧縮対象データ列を最後まで処理したか否かが判断されるが、この時点ではまだ終りでないから、ステップＲ６にて「ＮＯ」へ進み、ステップＲ２へ戻る。尚、ステップＲ４及びＲ５は、登録ステップに対応している。

【００２５】続いて、ステップＲ２においては、前回と同様にしてステップＳ１で最長一致データ長の初期化が行われた後、ステップＳ２に移行して、今度は第１アドレスの文字データ「Ｂ」と第２アドレスの文字データ「Ｃ」とでハッシングを行い、ハッシュ値0x01を得る。そして、インデックス0x01によってハッシュ表の検索を行うが、やはり”登録データなし”を検出し、次のステップＳ３で「ＮＯ」へ進み、ステップＲ３へ移行する。

【００２６】このステップＲ３においては、前回と同様

8

にして、出力データ列の第２アドレスに文字データ「Ｂ」をそのまま出力する。そして、ステップＲ４に移行し、ハッシュ表のインデックス0x01の領域に文字データ「Ｂ」の位置（アドレス）を示す位置情報「１」を書込んで登録する。また、圧縮対象データ列のポインタをインクリメントして次は第２アドレスの文字データ「Ｃ」に注目するようにしてから、次のステップＲ５に移行する。このステップＲ５では、１文字だけ処理したので「ＹＥＳ」へ進み、ステップＲ６に移行する。このステップＲ６では、データはまだ終りではないので「ＮＯ」へ進み、ステップＲ２に戻る。

【００２７】次の第３及び第４アドレスの文字列「ＣＤ」、また第４及び第５アドレスの文字列「ＤＡ」についても上記と同様に一致情報は出力されず、出力データ列には文字データ「Ｃ」及び「Ｄ」をそのまま出力し、ハッシュ表の各ハッシュ値のインデックスが示す領域には、位置情報「２」及び「３」がそれぞれ書込まれる。

【００２８】そして、圧縮処理対象データ列のポインタが「４」になると、ステップＳ２において第４及び第５アドレスの文字列「ＡＢ」に対してハッシングが行われる。すると、ハッシュ値0x03を得るので、ハッシュ表の検索を行うと登録データの位置情報「０」を得る（図６参照）。よって、次の判断ステップＳ３では「ＮＯ」へ進み、次の「同じデータ列か？」の判断ステップＳ４に移行する。

【００２９】ここで、ハッシュ法は大集合から小集合への写像を行うものであるから、異なるデータ列でも同一のハッシュ値になる場合（衝突）がある。本実施例では、同じハッシュ値を示す文字列の位置情報は、全て同一のインデックスに連なる連鎖的なデータ，所謂ハッシュチェーン（以下、単にチェーンと称す）に登録されるので、判断ステップＳ４においては、位置情報「０」のデータ列が現在の圧縮処理対象データ列と同じものであるかが確認される。そのため、判断ステップＳ４では、一致比較ロジック５によって、圧縮処理対象データ列の第０アドレスのデータ「Ａ」を読出して、第４アドレスのデータ「Ａ」と比較し、一致の結果を得る。ここで、同じハッシュ値において第１データが一致していれば第２データも必ず一致するので確認は終了する。そして、次の「１データずつ比較」の処理ステップＳ５に移行する。

【００３０】この処理ステップＳ５においては、文字列の第３データ以降を１データずつ比較するため、一致比較ロジック５により第２アドレスのデータ「Ｃ」及び第６アドレスのデータ「Ｃ」を読出して比較し、一致の結果を得る。次に第３アドレスのデータ「Ｄ」及び第７アドレスのデータ「Ｅ」を読出して比較すると不一致の結果を得るので、ここまでの一致データ長「３」を、メモリ３の現在の一致データ長の記憶領域に書込んで記憶させる。尚、ステップＳ４及びＳ５は、比較ステップに対

9

応している。

【００３１】続いて、「最長一致長か？」の判断を行う（ステップＳ６）。ここでは、選択ロジック７によって、ステップＳ５において得られた現在の一致データ長「３」が、最長一致データ長の領域に記憶されている内容より大きいか否かが判断される。この場合、最長一致データ長はステップＳ１で初期化されており「０」であるから、ステップＳ６にて「ＹＥＳ」へ進み、「一致位置と最長一致長を更新」の処理を実行する（ステップＳ７）。

【００３２】このステップＳ７においては、メモリ３上の一致位置の記憶領域に「０」を、最長一致データ長の記憶領域に「３」を書込んで記憶させる。そして、次の「チェーンの次をとる」の処理ステップＳ８に移行する。尚、ステップＳ６及びＳ７は、選択ステップに対応する。

【００３３】上記処理ステップＳ８においては、インデックス0x03のチェーンの次の（この場合、第１）データを読出す。すると、”登録データなし”のコードが読出されるので（図６参照）、次のステップＳ３では「ＮＯ」へ進み、ステップＲ３へ移行する。尚、ステップＳ２は検索ステップに対応し、ステップＳ３及びＳ８は繰返しステップに対応する。

【００３４】そして、ステップＲ３では、ステップＲ２の検索処理で得られた最長一致データ長が所定値「３」以上であるので、データ圧縮ロジック７によって、出力データ列の第５アドレスに一致情報として一致位置「０」及び一致データ長「３」を、例えば各１バイトサイズとした２バイトサイズのデータとして書込む。そして、第５アドレスが一致情報であることを示すため、第０アドレスの管理部に設定を行う。

【００３５】この管理部は、他の文字データと同じ１バイトサイズのデータであり、出力データ列の第０アドレスからデータ８個おきに挿入され、その内容は、その後に続く８個のデータの内何データ目がデータ圧縮処理の結果により出力された一致情報であるかを示すものである。例えば、この場合設定を行うには、第０アドレスの管理部の内容を読出すと、既に「１」が立てられているビットをクリアせずに新たな情報を付加するため、その内容と第４ビット（第５アドレスに対応する）のみ「１」を立てた１バイトデータとを排他的論理和により論理演算する。この時点では、管理部の内容はステップＲ１における初期設定時に０クリアされているので、排他的論理和により論理演算した結果は、図４（ｃ）に示すように第４ビットのみ「１」となる。そして、ステップＲ４に移行する。

【００３６】ステップＲ４においては、位置情報「４」をハッシュ表に登録する。この時点では、図６に示すように、ハッシュ表のインデックス0x03が示す領域には位置情報「０」が既に登録されているので、その位置情報

10

「０」をチェーンの第１領域に送出する。そして、ハッシュ表のインデックス0x03が示す領域に位置情報「４」を書込んで更新登録する。この更新登録後の状態を図７に示す。その後、圧縮処理対象データ列のポインタをインクリメントして「５」にすると、ループカウンタをインクリメントした後ステップＲ５に移行する。

【００３７】ステップＲ５においては、この時点での処理データ数は、ステップＳ５において選択ロジック６により得られた最長一致データ長が「３」であるから「３」と設定されており、ループカウンタは１であるから「ＮＯ」と判断して、この後を２回ループしてステップＲ４を２回実行する。即ち、文字列「ＢＣ」でハッシングしてその位置情報「５」と、文字列「ＣＥ」でハッシングしてその位置情報「６」とを、ハッシュ表のそれぞれのインデックスが示す領域に登録する。この処理の終了時点で、圧縮処理対象データ列のポインタは「７」である。そして、ステップＲ６では、データはまだ終了ではないので「ＮＯ」と判断して、ステップＲ２に移行する。

【００３８】ステップＲ２では、第７及び第８アドレスの文字列「ＥＡ」について、上記と同様な処理により、出力データ列への出力及びハッシュ表への登録を行う。以下はデータ列「ＡＢ」に注目して説明する。第８及び第９アドレスの文字列「ＡＢ」について処理を行う場合、ステップＳ２でハッシングしてハッシュ表の検索を行うと、位置情報「４」を得る（図７参照）。ステップＳ４及びＳ５において、第４アドレスから始まる文字列について１データずつ比較をおこなうが、得られる一致データ長は「２」である。従って、ステップＳ６では初期化された最長一致データ長「０」と比較して「ＹＥＳ」と判断して、ステップＳ７に移行する。そして、ステップＳ７において一致位置「４」及び最長一致データ長「２」を更新登録すると、ステップＳ８に移行する。

【００３９】ステップＳ８においては、インデックス0x03のチェーンの第１領域に格納されたデータを読出す。すると、位置情報「０」を得るので、ステップＳ３では「ＮＯ」と判断してステップＳ４に移行し、更に、ステップＳ４では「ＹＥＳ」と判断してステップＳ５に移行する。この場合も、ステップＳ５において得られる一致データ長は「２」であるので、以下同様にステップＳ６，Ｓ８，Ｓ９，Ｓ１０，Ｓ３と移行して、ステップＳ１０ではチェーンの第２領域に格納された”データなし”コードを得るので、ステップＳ３では「ＹＥＳ」と判断してステップＲ３に移行する。

【００４０】ステップＲ３においては、一致データ長が「２」の場合はデータ圧縮効果がないので、先頭の文字データ「Ａ」をそのまま出力データ列に出力する。そして、ステップＲ５でハッシュ表に登録されている位置情報「４」をチェーンに送出し、位置情報「８」をハッシュ表に登録する（図８参照）。以下、文字列「ＢＢ」及

び「ＢＡ」についても同様の処理を行う。

【００４１】次に、第１１及び第１２アドレスの文字列「ＡＢ」について、検索処理を行う。まず、ハッシュ表に登録された位置情報「８」について文字列の比較を行うと、３文字目で異なるので一致データ長「２」を得る。そして、次の検索処理でチェーンの第１領域に格納された位置情報「４」を得て文字列の比較を行い、やはり一致データ長「２」を得る。更に、次の検索処理でチェーンの第２領域に格納された位置情報「０」を得て文字列の比較を行う。すると、やはり一致データ長「２」を得る。更に次の検索処理では”データなし”を検出して検索処理を終了するが、最長一致データ長は「２」であるから、結果として圧縮処理は実行されない。

【００４２】以降、文字列「ＡＢ」，即ちインデックス0x03のチェーンは、図９及び図１０に示すように、入力データの位置情報「１１」と「１６」がハッシュ表に順次登録されることにより、位置情報データが順次チェーンに送出されて次第に長くなって行く。その中で最長の一致データ長が得られたものを一致情報として文字列の代わりに出力データ列に出力する。以下同様にして、データ圧縮処理を圧縮処理対象データの全てについて行うと、ステップＲ６で「ＹＥＳ」へ進み、「後処理」の処理ステップＲ７に移行する。このステップＲ７では、図４（ｂ）に示す圧縮処理後のデータであるメモリ３上の出力データを、データ入出力装置１によって外部記憶装置へ出力して、データ圧縮処理を終了する。

【００４３】以上のように本実施例によれば、制御回路２によって、圧縮処理対象のデータ中における圧縮処理済みデータの次の位置から始まる圧縮対象データ列と同じデータ列の候補を圧縮処理済みデータの中からハッシュ法によって検索すると共に、この検索されたデータ列と圧縮対象データ列とを比較して一致しているデータ長を求める構成とした。そして、制御回路２によって、圧縮対象データ列と同じデータ列の候補を圧縮処理済みデータの中から検索候補がなくなるまで上記検索及び比較を繰返し実行するように構成した。更に、制御回路２によって、上記比較により求められた一致データ長の中から最長のデータ長のデータ列を選択して出力すると共に、圧縮対象データ列をハッシュ法による検索候補としてハッシュ表に登録するように構成した。この構成によれば、圧縮処理対象データ列と同じデータ列の検索を行うのにハッシュ法を用いているので、データ列の検索が高速に実行できると共に、使用するメモリ空間を十分小さくすることができる。

【００４４】図１１は本発明の第２実施例を示すものであり、第１実施例と同一部分には同一符号を付して説明を省略し、以下異なる部分のみ説明する。図１１においては、第１実施例における図２のフローチャートのステップＳ７とステップＳ８との間に、「検索回数カウンタ＋１」の処理ステップＳ９及び「検索回数＝制限回数？」の判断ステップＳ１０を挿入するように構成されている。そして、第２の実施例の他の構成は、第１実施例と同じ構成である。

【００４５】次に、上記第２実施例の作用を説明する。第２実施例では、ステップＳ７の処理を終えると、次の「検索回数カウンタ＋１」の処理ステップＳ９に移行する。このステップＳ９においては、検索ロジック４によって一つのデータ列について検索表を１回検索すると、ステップＳ１で初期化された検索回数カウンタを１カウントアップして、次の「検索回数＝制限回数？」の判断ステップＳ１０に移行する。

【００４６】このステップＳ１０においては、検索回数カウンタのカウント値が、検索表の検索の制限回数として予め決められている値に等しくなったか否かが判断される。この場合、制限回数は「３」に設定されている。上記ステップＳ１０において、検索回数カウンタのカウント値が制限回数に等しくなると、「ＹＥＳ」へ進み、検索処理を完了する。また、ステップＳ１０において、検索回数カウンタのカウント値が制限回数に等しくないと（制限回数より小さいと）、「ＮＯ」へ進み、ステップＳ８に移行する。尚、第１実施例における繰返しステップにステップＳ９及びＳ１０を加えたものが、第２実施例の繰返しステップに対応する。

【００４７】ここで、第２実施例においても、第１実施例と同じ圧縮処理対象データについて圧縮処理を行うものとして、同様に文字列「ＡＢ」に注目して説明する。図７に示すように、ハッシュ値0x03のインデックスが示すハッシュ表に文字列「ＡＢ」の位置情報「４」が登録されている段階までは、ステップＳ９における検索回数カウンタのカウント値は「３」未満であり、ステップＳ１０においては「ＮＯ」と判断してステップＳ８に移行するので、第１実施例と同様に処理される。

【００４８】そして、図８に示す、次の位置情報「８」が登録された段階で、第１１及び第１２アドレスの文字列「ＡＢ」について検索を行うと、ステップＳ８においてチェーンの第２領域にある位置情報「０」を読出した後、ステップＳ３乃至Ｓ７まで同様に処理を行い、次のステップＳ９において検索回数カウンタのカウント値は制限回数である「３」に達する。すると、ステップＳ１０にて「ＹＥＳ」へ進むから、それ以上の検索及び比較を行うこと無く検索処理を打切るように構成されている。

【００４９】また、図９及び１０に示すように、その後にデータ圧縮処理が進んで位置情報「１１」と「１６」がハッシュ表に順次登録されることによりチェーンが更に長くなった場合でも、この後に圧縮処理対象データ列中に文字列「ＡＢ」が出現した場合は、ステップＲ２での検索処理は３回だけ，即ち、位置情報「１６」，「１１」及び「８」についてのみ行ってそれ以上の検索表の検索は行なわず、その中で最長の一致データ長が得られ

たものを一致情報として出力データ列に出力するように構成されている。

【００５０】尚、第２実施例では、検索表内に出現している一致データ列全てについて検索を行わないため、最長の一致データ長が得られない可能性がある。しかし、一般的な文章データの特徴として、「最長の一致長を有するデータ列は、注目している位置の直ぐ近くに現れやすい」という性質がある。従って、第２実施例のように、最新の一致データ列から３回だけ検索する方式を用いても、圧縮効率の劣化は殆ど無いのである。

【００５１】以上のように第２実施例によれば、検索処理を行う回数をカウントして、予め３回として設定された制限回数に等しくなると検索を打切り、その時点で最長の一致データ長を示したデータ列の先頭位置とその一致データ長とを一致情報として出力するように構成したので、一致データ列がなくなるまで検索表の検索を反復する第１実施例とは異なり、検索に要する時間はチェーンが長くなった場合でも毎回同じとなり、データ圧縮処理をより一層高速に実行することができる。しかも、この場合、データ圧縮効率を低下させることはほとんどない。

【００５２】尚、本発明は上記し且つ図面に記載した実施例にのみ限定されること無く、次のような変形が可能である。まず、文字データの１６進数コードをＪＩＳコードとしたが、ＥＢＣＤＩＣコード若しくはＡＳＣＩＩコードでも良い。また、ハッシュ関数を排他的論理和としたが、これに限らず、衝突の発生がなるべく少なくなるような適当な演算（論理演算に限らない）を適宜選択するように構成しても良い。更に、ステップＲ３において、一致情報を出力する最長一致データ長の所定値を「３」としたが、「４」以上の適当な値に設定しても良い。加えて、ハッシングの対象とする文字数を３文字以上としても良い。

【００５３】また、一致情報のデータ（２バイトのデータ）は、一致位置（アドレス）を表わす１バイトのデータと、一致データ長を表わす１バイトのデータとから構成したが、これに限らず、圧縮対象データの大きさと予想される最大一致データ長との兼合いによって適宜変更しても良く、例えば一致位置のサイズを１２ビット、一致データ長のサイズを４ビットとしても良い。更に、一致情報のデータサイズは２バイトに限らず、圧縮処理対象データの大きさに応じてビット数が更に必要な場合は、例えば３バイトとしても良い。この場合、その一致情報のデータサイズに応じて圧縮効率を考慮し、一致情報を出力する最低の一致データ長を「３」から「４」にするなど、適宜変更して良い。

【００５４】更にまた、圧縮処理対象データは、アルファベットなどからなる言語の文章に限らず、日本語などのその他の言語でも良い。また、言語の１文字を表現する１６進数コードが２バイトである場合は、管理部を、そのデータサイズを２バイトとして、１６文字毎に１個付加するように構成しても良い。

【００５５】また、反復ロジックにおける検索の制限回数（打切り回数）を３回としたが、この回数は、データ圧縮効率と処理速度とを考慮して、どちらをより重視するかによって適宜決定すれば良い。例えば、圧縮効率を重視する場合は検索制限回数を大きな値に設定し、処理速度を重視する場合は検索制限回数を小さな値に設定する。データ圧縮効率と処理速度との適当なバランスを重視して決定する場合は、検索制限回数は３乃至２０回の何れかに設定することが望ましい。この場合、最も好ましい制限回数は、１０回前後である。また、圧縮処理対象とするデータの性質に応じて、ユーザーが検索の制限回数を指定可能にするように構成しても良い。

【００５６】一方、上記各実施例においては、最新の圧縮対象データ列をハッシュ表の最初に検索される位置に登録するように構成したが、これに限られるものではなく、最新の圧縮対象データ列を一番最後に検索される位置に登録するように構成しても良い。

【００５７】また、上記各実施例のステップＳ５において、限界として規定される最長一致データ長（最長一致データ長の記憶領域の物理的な制限，若しくは、文章データの性質からこれ以上の一致データ長は有り得ないと想定されるもの）を検出した場合は、それ以降の検索を打切るように構成しても良い。この構成によれば、更に圧縮処理を高速にすることができる。

【００５８】更に、図１１のフローチャートの「リターン」の直前に、「検索回数を記憶」の処理ステップＳ１１を設け、データ列の検索毎に、最長の一致長が検出されたデータ列が何回目の検索で見付かったか、その検索回数をメモリ３に記憶させるように構成する。そして、データ圧縮処理がある程度進んだ段階で、「検索回数最適化処理」の処理ルーチンを実行するように構成しても良い。この処理ルーチンにおいては、ステップＳ１１で記憶させた検索回数に対して適当な統計処理を行うことによって、圧縮効率が十分良い範囲で且つ検索処理時間を十分短縮できる検索回数を自動的に求めて設定する（最適化する）ことができるように構成されている。

【００５９】尚、上記各実施例や変形例等によって圧縮処理して出力した出力データに対して、例えば算術符号化などの他の圧縮方法を適用して更にデータ圧縮するように構成しても良く、このように構成すると、更にデータ圧縮効率を高めることができる。

【００６０】

【発明の効果】本発明は、以上の説明から明らかなように、ＬＺスライド辞書法を用いて文書データ等のデータを圧縮するデータ圧縮処理において、圧縮対象データ列と同じデータ列を圧縮処理済みデータの中からハッシュ法によって検索する構成としたので、データ列を高速で検索することができると共に、検索時に必要なメモリ空

15

間を極力少なくすることができる。

【００６１】また、この構成の場合、検索及び比較を所定回数繰返した後、打切るように構成すると、データの検索及びそれに伴うデータ圧縮処理をより一層高速に実行できる。更に、検索を打切る回数を、３乃至２０回の何れかに設定すると、データの圧縮効率と圧縮処理速度との適度なバランスをとることができる。そして、最新の圧縮対象データ列をハッシュ法による検索候補のうちの最初に検索される位置に登録するように構成すると、最長の一致データ長を得る確率を高めることができる。また、最長の一致データ長を検出したときの検索回数を統計的に記憶することにより、検索を打切る回数を自動的に最適化して決定するように構成すると、データ圧縮効率及び圧縮処理速度を最適化することができる。

【図面の簡単な説明】

【図１】本発明の第１実施例の構成を示すブロック図

【図２】制御内容のフローチャート

【図３】図２のステップＲ２の制御内容のフローチャート

16

【図４】入力データ及び出力データ並びに管理部の内容を示す図

【図５】初期化された状態の検索表の内容を示す概念図

【図６】ハッシュ表の更新登録処理を示す概念図

【図７】図６相当図

【図８】図６相当図

【図９】図６相当図

【図１０】図６相当図

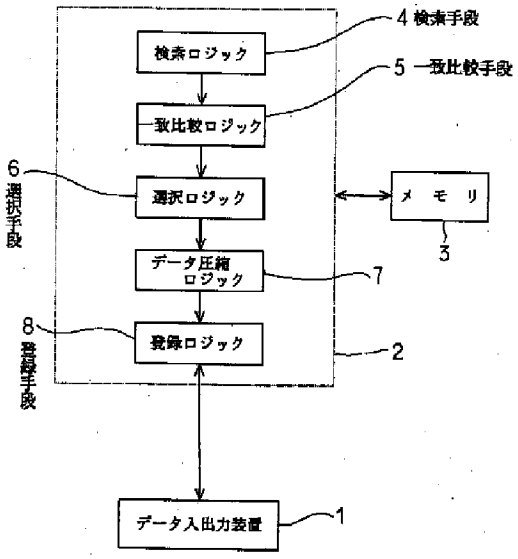
【図１１】本発明の第２実施例を示す図２相当図

【図１２】ＬＺスライド辞書法によるデータ圧縮処理の概念図

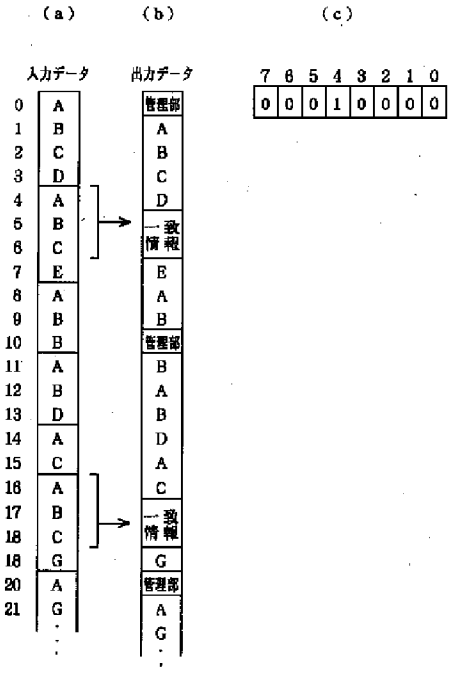
【符号の説明】

１はデータ入出力装置、４は検索ロジック（検索手段）、５は一致比較ロジック（比較手段）、６は選択ロジック（選択手段）、７はデータ圧縮ロジック、８は登録ロジック（登録手段）を示す。

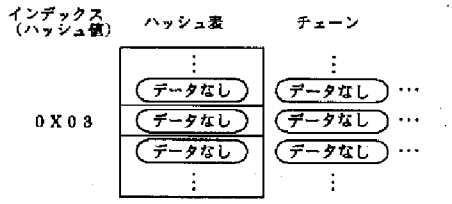
【図１】



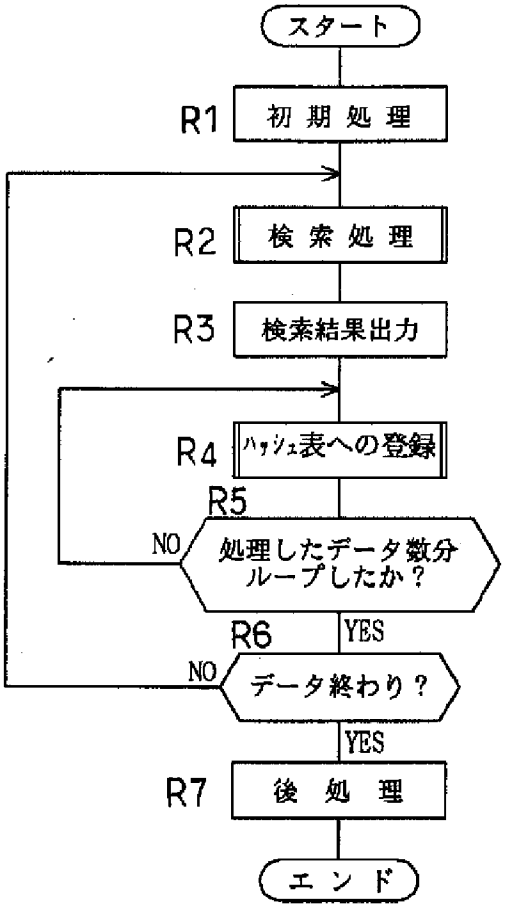
【図４】



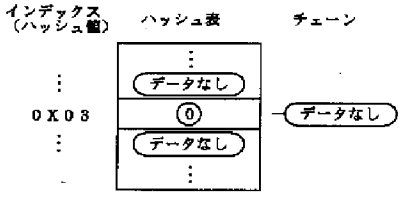
【図５】



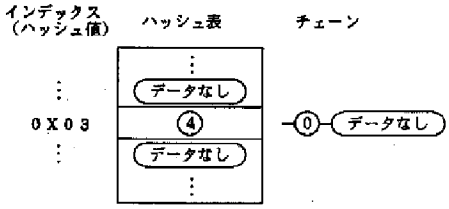
【図２】


スタート
R1
初期処理
R2
検索処理
R3
検索結果出力
R4
ﾊｯｼｭ表への登録
R5
NO
処理したデータ数分ループしたか？
R6
YES
NO
データ終わり？
YES
R7
後処理
エンド

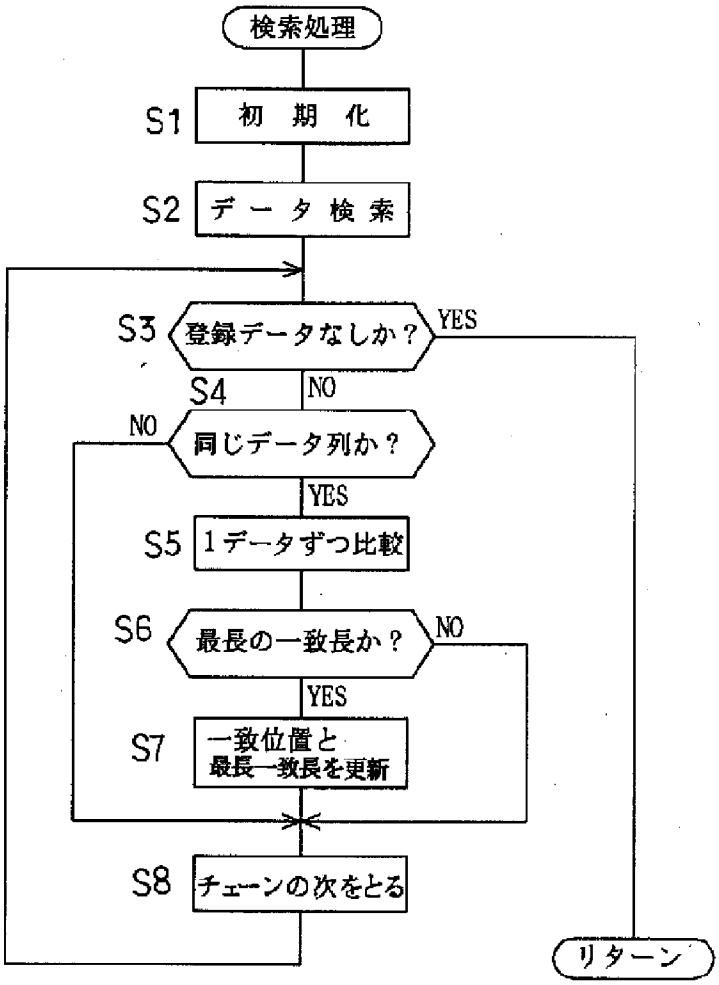
【図６】


インデックス
(ハッシュ値)
ハッシュ表
チェーン
0X03
データなし
0
データなし
データなし

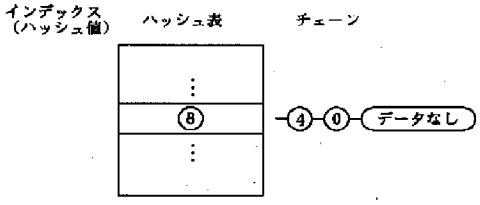
【図７】


インデックス
(ハッシュ値)
ハッシュ表
チェーン
0X03
データなし
4
データなし
0
データなし

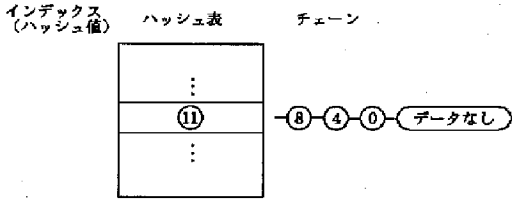
【図３】


検索処理
S1
初　期　化
S2
デ　ー　タ　検　索
S3
登録データなしか？
YES
S4
NO
NO
同じデータ列か？
YES
S5
1データずつ比較
S6
最長の一致長か？
NO
YES
S7
一致位置と
最長一致長を更新
S8
チェーンの次をとる
リターン

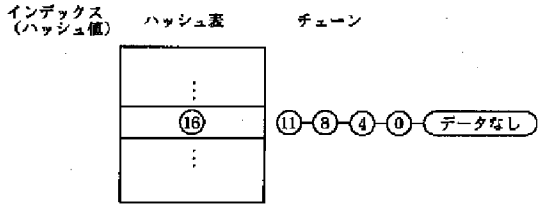
【図８】


インデックス
(ハッシュ値)
ハッシュ表
チェーン
⑧
④
⓪
データなし

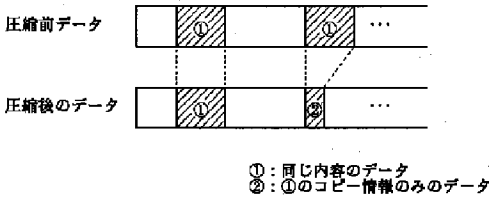
【図９】


インデックス
(ハッシュ値)
ハッシュ表
チェーン
⑪
⑧
④
⓪
データなし

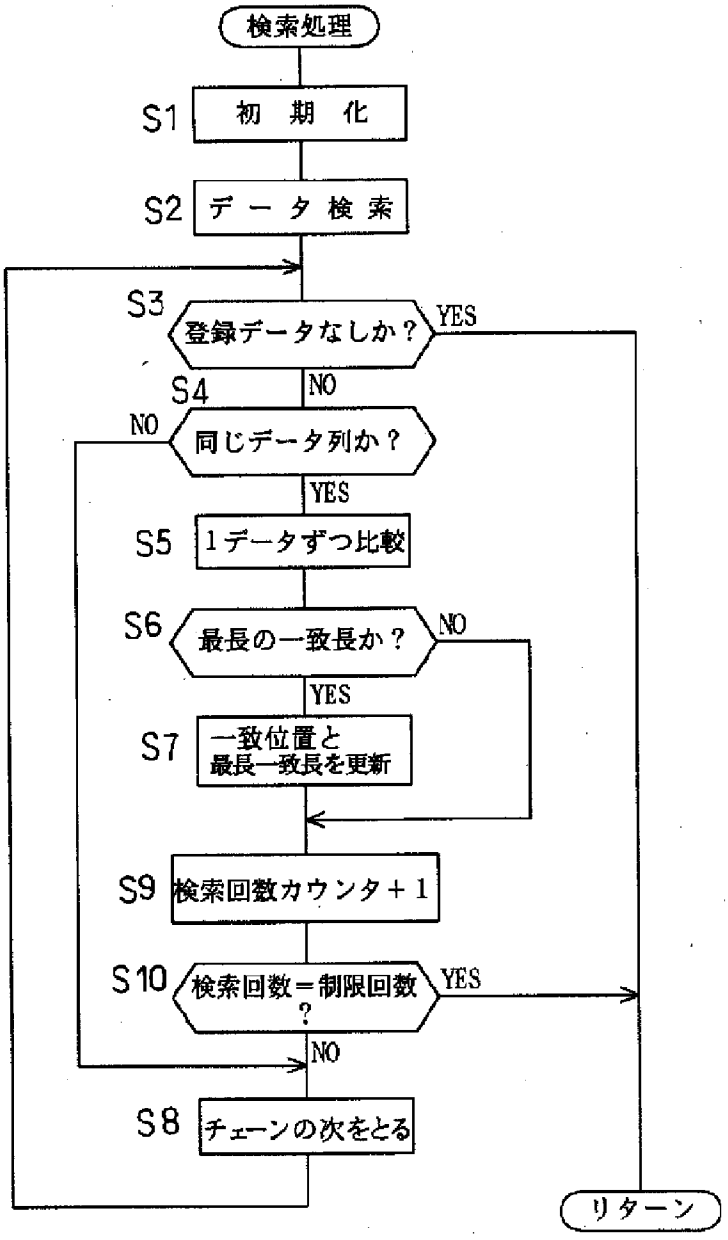
【図１０】


インデックス
(ハッシュ値)
ハッシュ表
チェーン
16
11
8
4
0
データなし

【図１２】


圧縮前データ
圧縮後のデータ
①
①
①
②
…
…
①：同じ内容のデータ
②：①のコピー情報のみのデータ

【図１１】


検索処理
S1
初　期　化
S2
デ　ー　タ　検　索
S3
登録データなしか？
YES
NO
S4
同じデータ列か？
NO
YES
S5
1データずつ比較
S6
最長の一致長か？
NO
YES
S7
一致位置と
最長一致長を更新
S9
検索回数カウンタ＋1
S10
検索回数＝制限回数
？
YES
NO
S8
チェーンの次をとる
リターン